

**SOFTWARE ENGINEERING**

**CSE 1005**

**LAB - L35 + L36**

**AI BASED PERSONAL FINANCE MANAGER**

**(MACHINE LEARNING)**

**SRS DOCUMENT**



**VIT-AP**  
**UNIVERSITY**

**BATCH-3**

**Submitted by**

**N. BHUVANESH 22BCE7072**

**N. CHARAN SANJAY 22BCE7811**

**K. V. PAVAN KUMAR 22BCE9481**

**V. KRISHNA CHAITANYA 22BCE9530**

**B. DHEERAJ NAGA ABHISHEK 22BC9925**

**Faculty**

**Prof. Shaik Kareemulla**

# AI BASED PERSONAL FINANCE MANAGER

## **Problem Statement :-**

Personal finance management has gotten more difficult and time-consuming in today's busy society. People often find it difficult to keep a clear and ordered picture of their financial health because of their numerous income sources, various expenses, investments, savings objectives, and debt management. Standard financial tracking techniques, such as manual spreadsheets or simple budgeting applications, frequently fail to deliver specific, instant insights and suggestions.

The issue gets worse by the fact that a large number of people lack financial literacy, which results in bad decisions, underutilized money, and lost investment possibilities. Furthermore, given the wide range of financial behaviors and goals that users have, a solution that can adjust to each user's needs and preferences and provide personalized advice in line with those goals is necessary. So this Personal manager can give a clear layout of previous finance structures and future recommendations as well.

## **Table of Contents:**

### **1. Introduction**

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms, and Abbreviations

1.4 References

1.5 Overview

### **2. Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 Assumptions and Dependencies

### **3. Specific Requirements**

3.1 Functional Requirements

3.2 Non-Functional Requirements

## **4. Appendices**

## **1. Introduction**

### **1.1 Purpose**

The Software Requirements Specification (SRS) for an AI-based Personal Finance Manager application is intended to be defined by this document. Giving people smart tools and insights to manage their money effectively is the system's main objective. Utilising machine learning techniques, the program seeks to automate financial forecasting, expenditure analysis, investment recommendations, and budgeting. To ensure that the program is developed and deployed successfully, this document contains the functional and non-functional requirements, scope, definitions, and overall system architecture.

### **1.2 Scope**

By tracking expenses, making budgets, analysing spending patterns, and offering useful insights to help users make better financial decisions, this AI-powered personal finance manager is intended to help users manage their personal money. It will classify transactions, forecast future expenditure, integrate with several bank accounts, and make optimisation recommendations. Users will be able to monitor their finances at any time and from any location thanks to the application's availability via both mobile and online applications.

Important characteristics consist of:

**Automated Transaction Categorisation:** Classifying user transactions based on machine learning.

**Help with Budgeting:** Budgets are automatically created using historical expenditure trends.

**Expense tracking:** Monitoring and reporting costs in real-time.

Investment Advice: Tailored investment recommendations according to the user's financial objectives and risk tolerance. Using predictive analytics, financial forecasting projects future financial situations.

Security: To safeguard user financial information, secure data management and encryption are used.

### 1.3 Definitions, Acronyms, and Abbreviations

- **AI:** Artificial Intelligence
- **ML:** Machine Learning
- **API:** Application Programming Interface and **NLP:** Natural Language Processing

### 1.4 References

<https://www.kaggle.com/datasets/tharunprabu/my-expenses-data>

### 1.5 Overview

The document is set up to give readers a thorough grasp of the Personal Finance Manager powered by AI. After this introduction, Section 2 will describe the features of the system and the needs of the user and the system. The system architecture, including data flow diagrams and system models, will be covered in Section 3. Performance, security, and usability considerations are among the non-functional criteria that will be covered in Section 4. Lastly, other factors like risk management and potential improvements are covered in Section 5.

## **2.Overall Description**

### 2.1 Product Perspective

The goal of the AI-powered Personal Finance Manager is to provide customers with a comprehensive toolkit for efficient money management. By using artificial intelligence, it offers personalized financial advice, assistance with budgeting, expense tracking, and investment recommendations. The product stands out due to its powerful AI capabilities and user-centric features, even if it is a part of a wider ecosystem of financial management solutions.

## 2.2 Product functions

1. Data Collection and Preprocessing: Gather and preprocess financial data.
2. Analyze Spending Patterns: Apply machine learning models to examine user spending behaviors.
3. Personalized Budgeting: Create recommendations for a user's budget that are specific to their spending habits.
4. Optimize Investments and Savings: Apply AI algorithms to examine user investment and savings portfolios.
5. Expense Prediction and Forecasting: Based on past data, use predictive models to project future expenses.
6. Plan and Track Financial Goals: Help customers define attainable financial objectives, such as home ownership and retirement planning.
7. Anomaly Detection and Alerts: To identify odd or suspicious transactions, apply machine learning models.

## 2.3 User Classes And Characteristics

1. Data scientists: should create and improve machine learning models.
2. Developers: Put various system components into practice and integrate them.
3. Financial Analysts: Keep an eye on the system and offer financial analysis.
4. End Users (Clients): Manage your finances with the help of the personal financial manager.
5. Product Managers: Be in charge of the personal financial manager's development and implementation.

## 2.4 Operating Environment:

- Hardware: To facilitate intensive computation and massive data processing, servers with high-performance CPUs and GPUs are used.
- Software: TensorFlow, PyTorch, Flask, Docker, R, Scikit-learn, Python, and Keras

- Platforms: Google Colab, AWS, Azure, Jupyter Notebook
- Databases: NoSQL databases (like MongoDB) and SQL databases (like PostgreSQL, MySQL)
- Tools: Git and other version control systems
- Tools for Continuous Deployment and Integration (CI/CD)

## 2.5 Design and Implementation Constraints:

1.Data Privacy: To safeguard user data and privacy, make sure that all data protection laws, such as the GDPR, are followed.

2.Scalability: Without sacrificing performance, the system must be able to manage massive amounts of user data and transactions in real-time.

3.Security: Put strong security measures in place to guard against cyberattacks, illegal access, and data breaches.

4.Integration: To retrieve and process financial data, the system must be able to easily integrate with a variety of financial institutions and outside services.

5.User Experience: Make sure that users with different levels of financial understanding can easily navigate the system's intuitive and user-friendly interface.

## 2.6 Assumptions and Dependencies

1. Availability of transacted data which is labeled.
2. Financial institutions cooperation for data sharing
3. Steady and Expandable Cloud Framework
- 4.Consent from Users and Privacy Adherence
- 5.Combining with Current Financial Systems

## **3.Specific Requirements**

### 3.1 Functional Requirements

- FR1: User Account Management: The system shall allow users to create, manage, and delete their accounts.

- FR2: Financial Data Aggregation: The system shall aggregate financial data from various sources, including bank accounts, credit cards, and investment accounts.
- FR3: Budgeting and Expense Tracking: The system shall provide tools for budgeting and tracking expenses.
- FR4: Financial Planning and Forecasting: The system shall offer financial planning and forecasting tools.
- FR5: Investment Management: The system shall support investment management and provide recommendations.
- FR6: Tax Management: The system shall assist with tax estimation and filing support.
- FR7: Reporting and Analytics: The system shall provide reporting and analytical tools for financial data.
- FR8: Real-Time Data Processing: The system shall process financial data in real-time.
- FR9: Security and Privacy: The system shall ensure the security and privacy of user data.
- FR10: Integration and API Access: The system shall offer APIs for integration with other financial tools and services.

### 3.2 Non-Functional Requirements

- NFR1: Data Privacy and Compliance
  - The system shall ensure the highest level of data privacy and comply with all relevant data protection regulations, including the General Data Protection Regulation (GDPR).
- NFR2: Scalability: The system shall be scalable to handle up to 10,000 transactions per second without performance degradation.
- NFR3: Availability: The system shall be available 99.99% of the time, ensuring continuous access for users.
- NFR4: Mean Time to Recovery (MTTR): The system shall have a mean time to recovery (MTTR) of less than 1 hour in the event of any failures or service interruptions.
- NFR5: Logging and Monitoring: The system shall provide detailed logging and monitoring capabilities to track operations, detect anomalies, and facilitate auditing.

- NFR6: Performance: The system shall respond to user interactions and process transactions within a maximum of 200 milliseconds under normal operating conditions.
- NFR7: Usability: The system shall be user-friendly, with an intuitive interface that can be easily navigated by users with varying levels of financial literacy.
- NFR8: Security: The system shall implement security measures to protect against unauthorized access, data breaches, and cyber threats.
- NFR9: Maintainability: The system shall be designed for easy maintenance and updates, allowing for quick identification and resolution of issues.
- NFR10: Interoperability: The system shall be capable of integrating with third-party financial services and tools, such as banks, investment platforms, and payment processors.
- NFR11: Accessibility: The system shall be accessible to users with disabilities, complying with relevant accessibility standards such as WCAG 2.1.

## **4. Appendices**

### 4.1 Glossary

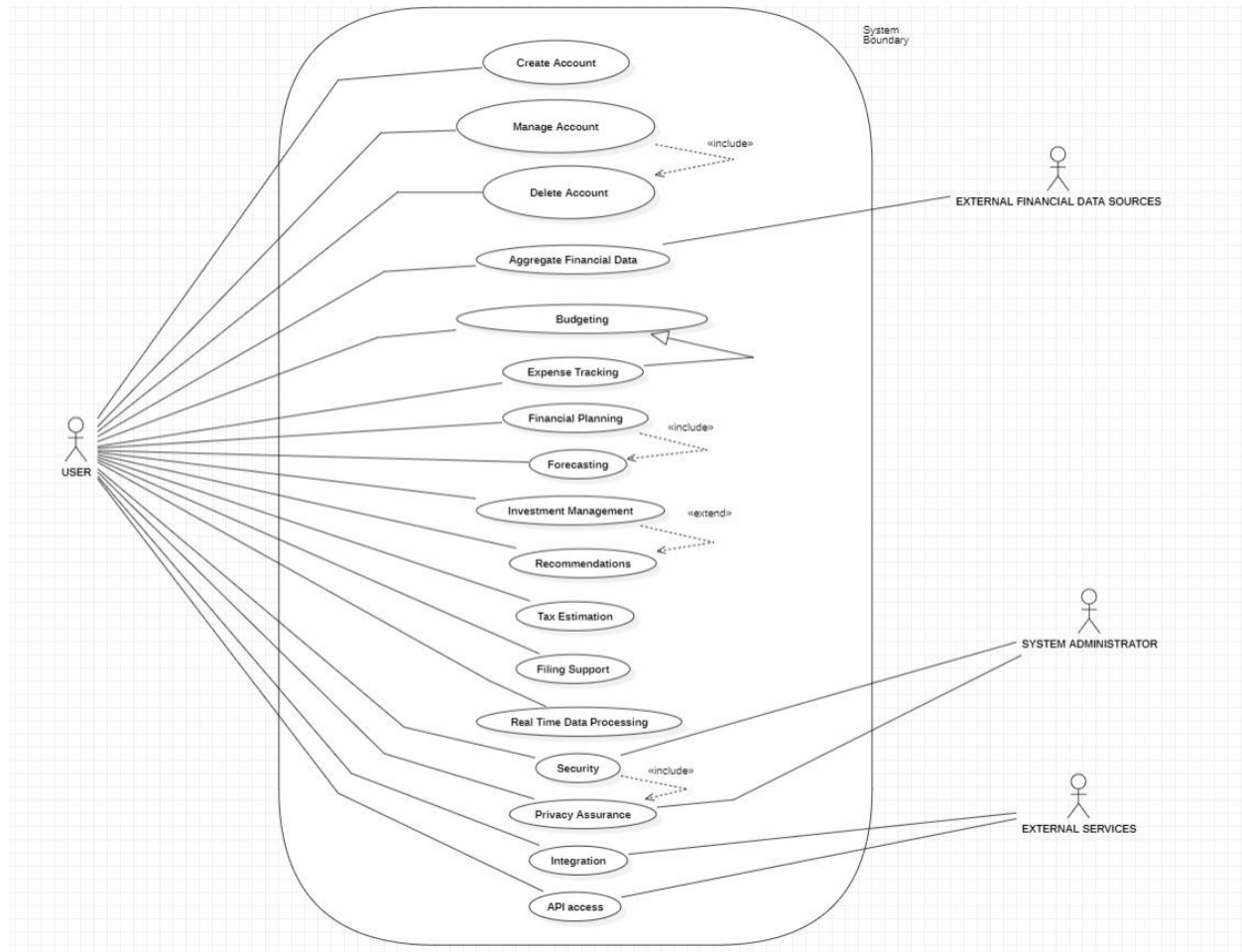
- Finance Manager : It deals with finance data and gives us predictions with better accuracy
- Machine Learning Model: An algorithm trained on data to make predictions.

### 4.2 References

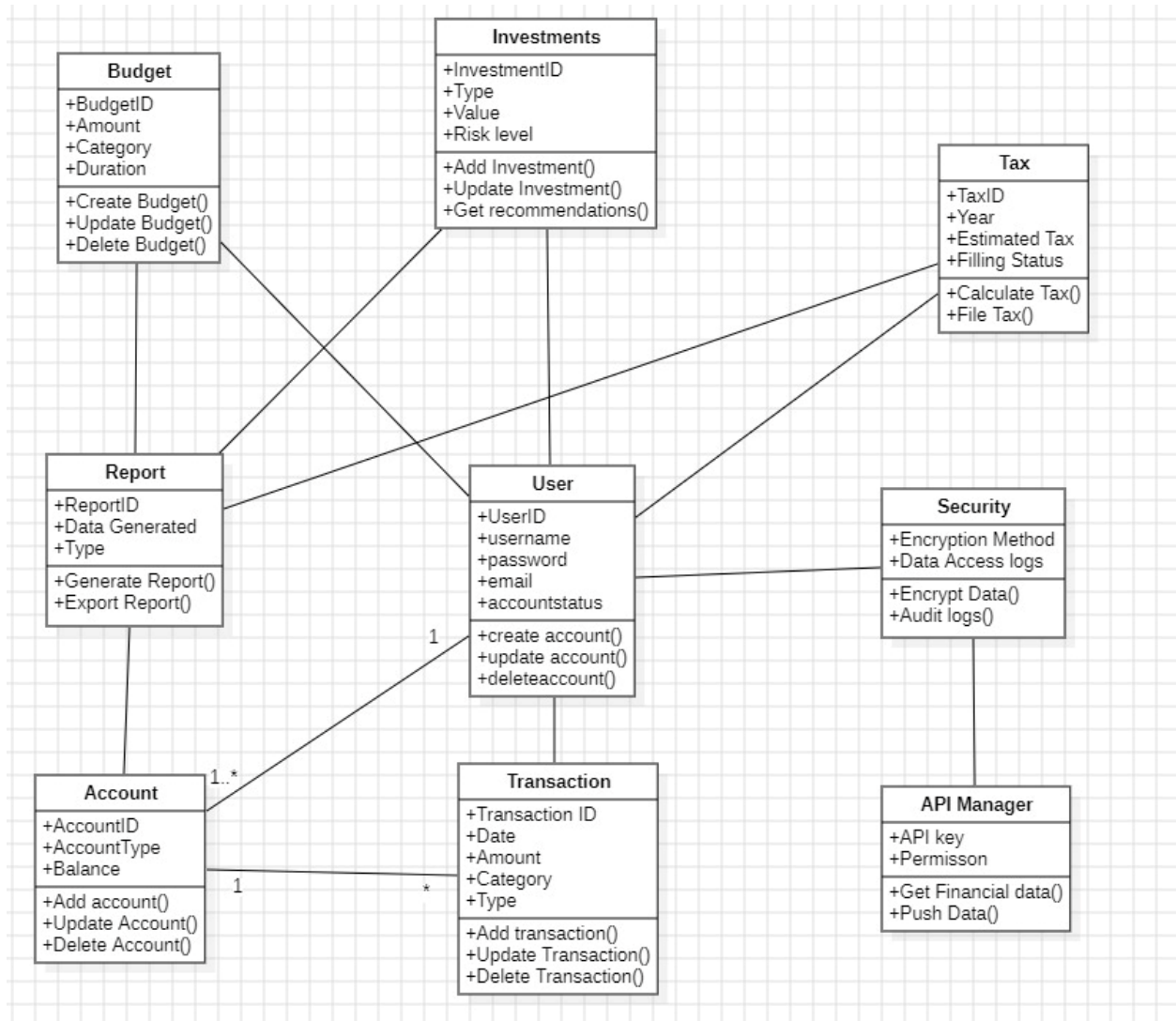
- Scikit-learn: <https://scikit-learn.org>
- TensorFlow: <https://www.tensorflow.org>
- Keras: <https://keras.io>



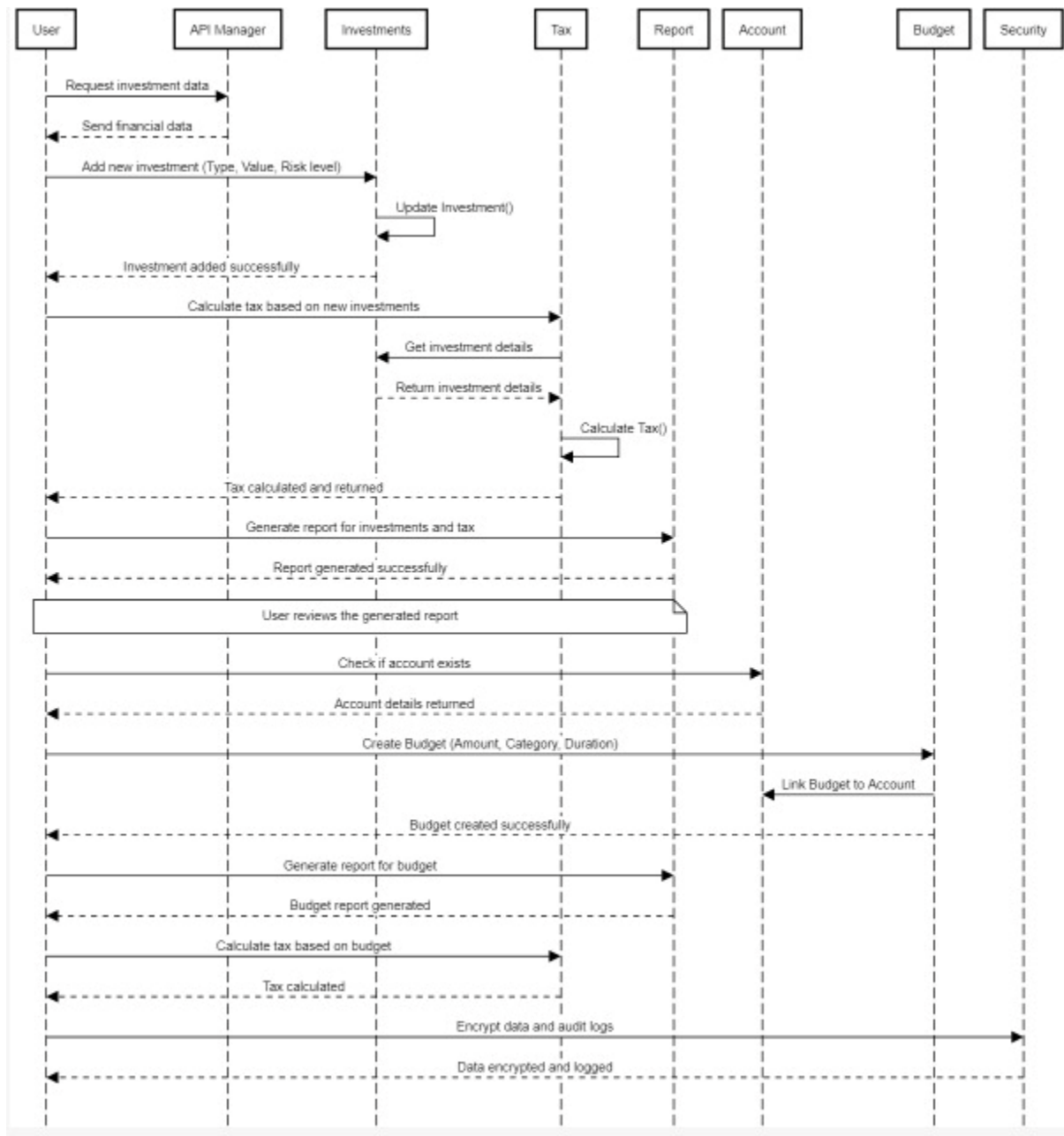
## USE-CASE DIAGRAM:-



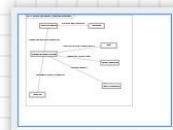
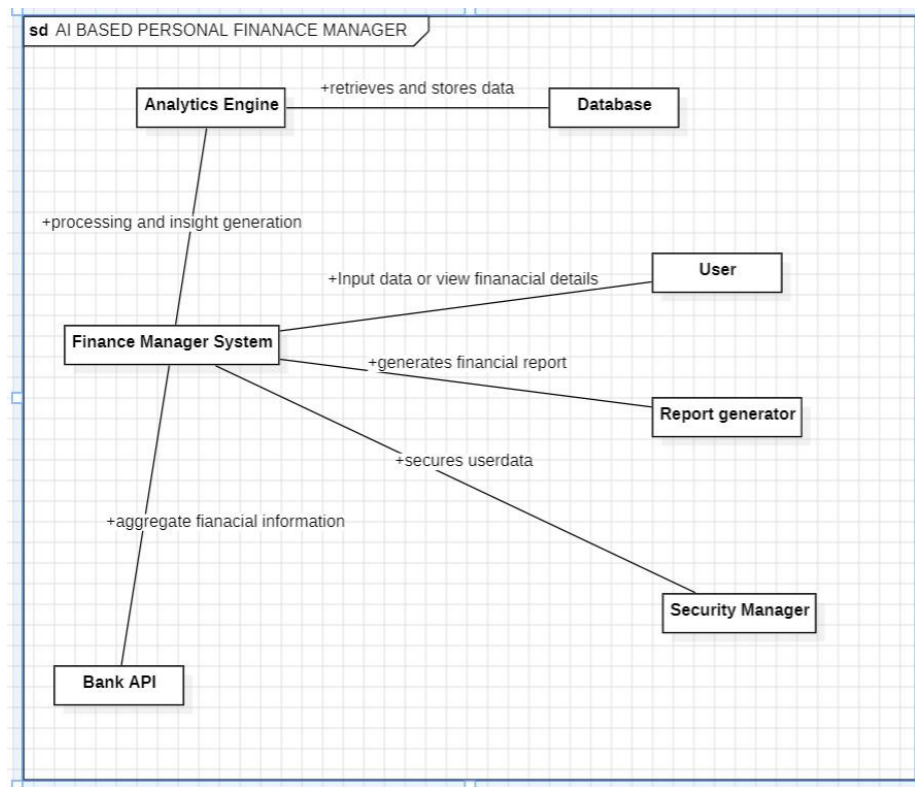
## CLASS DIAGRAM:-



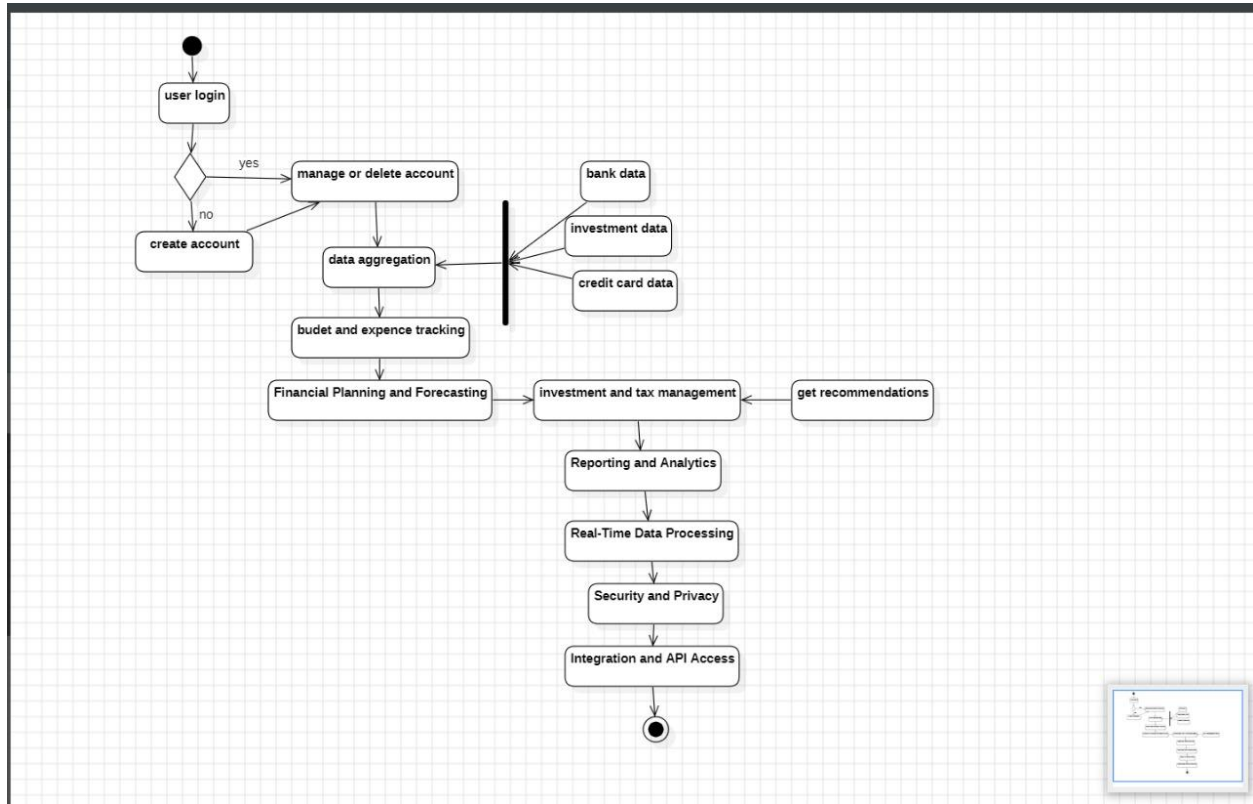
## SEQUENCE DIAGRAM:-



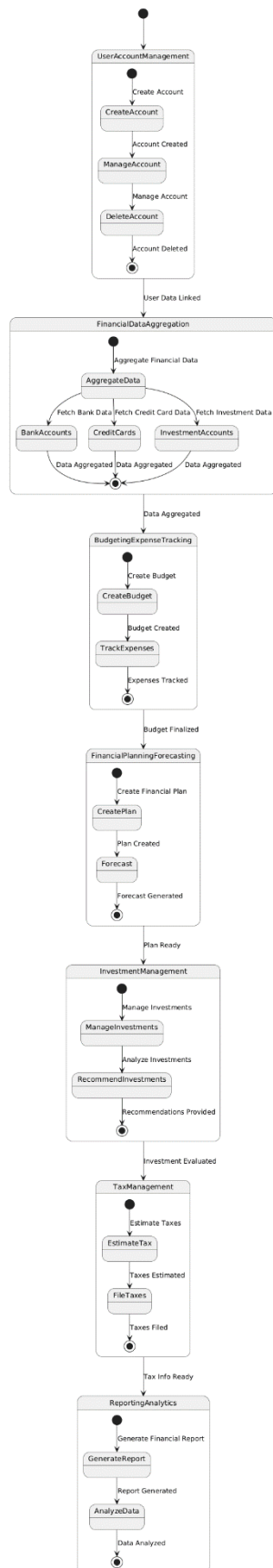
## COLLABORATION DIAGRAM:-



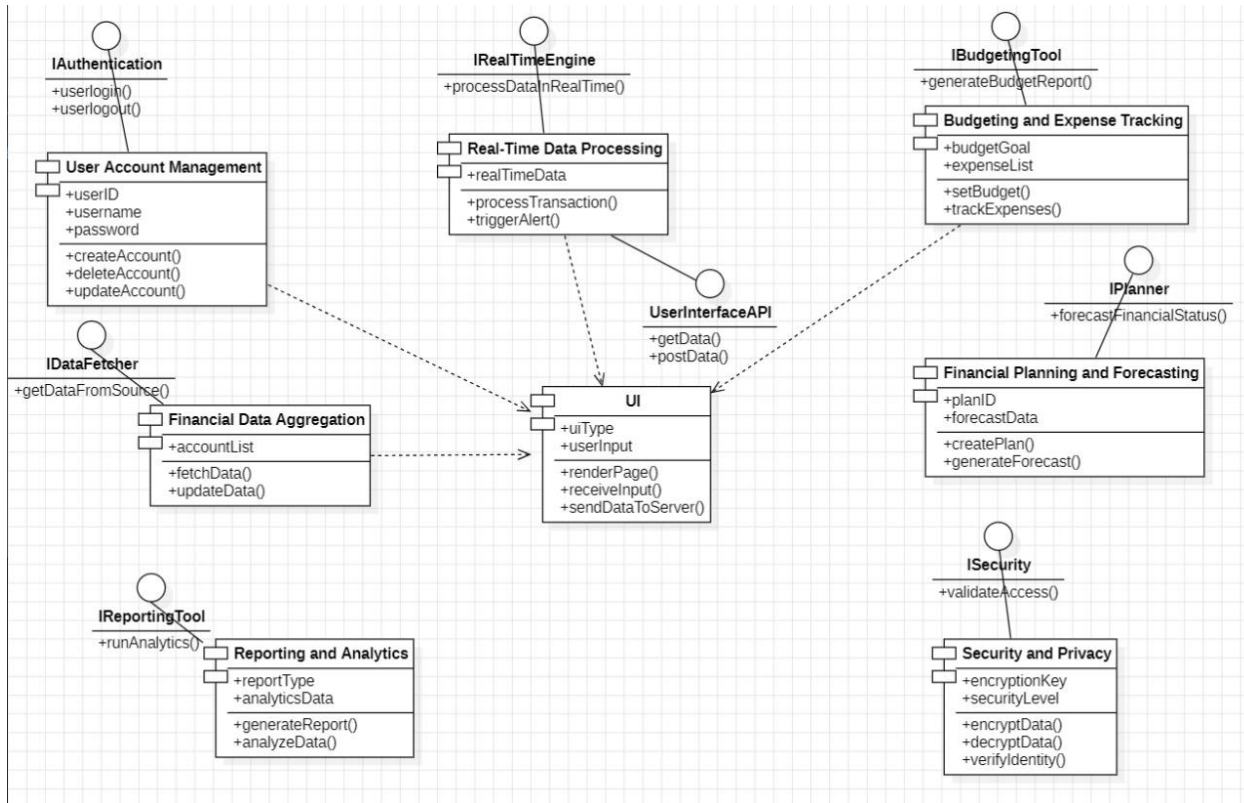
## ACTIVITY DIAGRAM:-



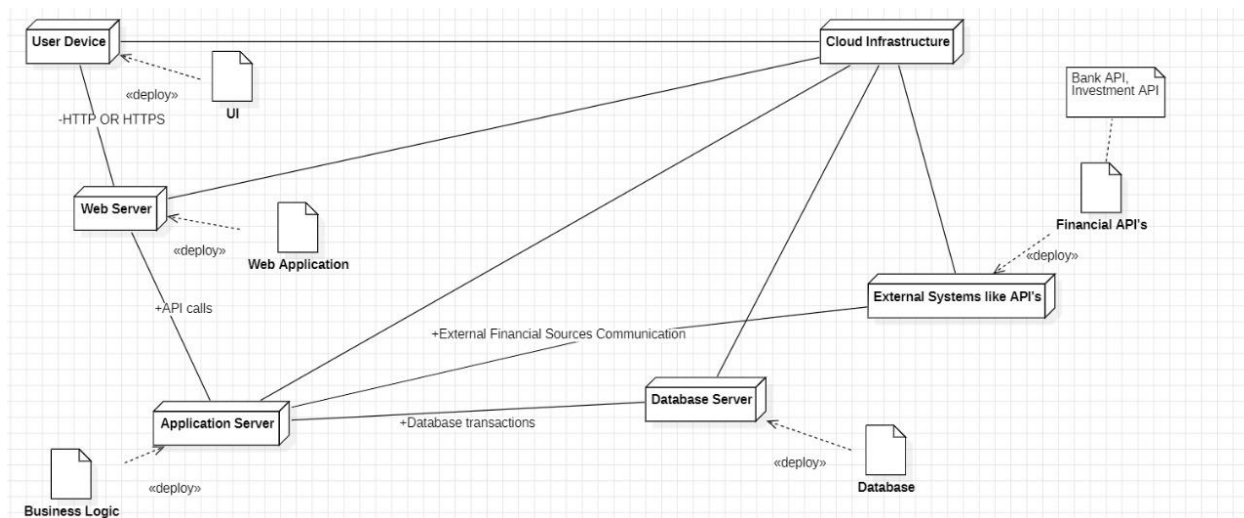
**STATE DIAGRAM:-**



## COMPONENT DIAGRAM:-



## DEPLOYMENT DIAGRAM:-





## **WEB-PAGES DESIGNING :-**

### **1.LOGIN PAGE:-**

#### **CODE :-**

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Login - AI-Based Personal Finance Manager</title>

  <style>

    body{

      font-family: Arial, sans-serif;

      background-color: white;

      margin: 0;

      padding: 0;

      display: flex;

      justify-content: center;

      align-items: center;

      height: 100vh;

      color: black;

    }

    .login-container {

      background-color: #f0f0f0;

      padding: 20px;

      border-radius: 10px;

      box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
```

```
    width: 300px;
    text-align: center;
}
.login-container input[type="text"],
.login-container input[type="password"] {
    width: 100%;
    padding: 10px;
    margin: 10px 0;
    border: 1px solid #ccc;
    border-radius: 5px;
    background-color: white;
    color: black;
}
.login-container button {
    padding: 10px 15px;
    margin: 5px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
}
.submit-button {
    background-color: #555;
    color: white;
    float: right;
}
.register-button {
```

```
background-color: #007bff;

color: white;

float: left;
}

.captcha {

margin: 15px 0;

display: flex;

justify-content: center;

align-items: center;
}

.captcha-box {

background-color: #e0e0e0;

padding: 10px 15px;

font-size: 18px;

font-weight: bold;

margin-right: 10px;

border: 1px solid #ccc;

color: black;
}

.refresh-captcha {

cursor: pointer;

color: blue;

text-decoration: underline;

margin-left: 10px;
}

.error-message {
```

```
        color: red;

        margin-top: 10px;
    }

    .button-container {

        display: flex;

        justify-content: space-between;
    }
</style>
</head>
<body>

    <div class="login-container">

        <h2>Login</h2>

        <form id="loginForm" action="#" method="POST">

            <input type="text" id="username" name="username" placeholder="Email" required>

            <input type="password" id="password" name="password" placeholder="Password"
required>

            <div class="captcha">

                <div class="captcha-box" id="captchaText">Loading...</div>

                <button type="button" class="refresh-captcha"
onclick="generateCaptcha()">Refresh</button>

            </div>

            <input type="text" id="captchaInput" name="captcha" placeholder="Enter Captcha"
required>

            <div class="error-message" id="errorMessage" style="display:none;"></div>

            <div class="button-container">

                <button type="button" class="register-button"
onclick="window.location.href='registration.html';">Register</button>
```

```
        <button type="submit" class="submit-button">Login</button>

    </div>

</form>

</div>

<script>

    function generateCaptcha() {

        const characters =
'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';

        let captcha = "";

        for (let i = 0; i < 6; i++) {

            captcha += characters.charAt(Math.floor(Math.random() * characters.length));

        }

        document.getElementById('captchaText').innerText = captcha;

    }

    window.onload = function() {

        generateCaptcha();

    };

    document.getElementById('loginForm').addEventListener('submit', function(event) {

        event.preventDefault();

        const username = document.getElementById('username').value;

        const password = document.getElementById('password').value;

        const captchaInput = document.getElementById('captchaInput').value;

        const captchaText = document.getElementById('captchaText').innerText;

        const errorMessageDiv = document.getElementById('errorMessage');

        errorMessageDiv.style.display = 'none';

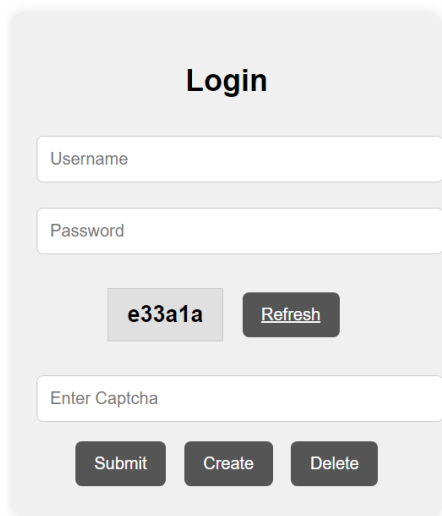
        errorMessageDiv.textContent = "";

    });

});
```

```
if (!username.includes('@') || password === "") {  
    errorMessageDiv.textContent = 'Invalid email/password.';  
    errorMessageDiv.style.display = 'block';  
    return;  
}  
  
if (password !== 'correct_password') { // Replace 'correct_password' with your actual  
password check  
    errorMessageDiv.textContent = 'Invalid username/password.';  
    errorMessageDiv.style.display = 'block';  
    return;  
}  
  
if (captchaInput !== captchaText) {  
    errorMessageDiv.textContent = 'Invalid captcha.';  
    errorMessageDiv.style.display = 'block';  
    return;  
}  
  
alert(` Welcome, ${username}! You have logged in successfully.` );  
  
    // Redirect to home page or perform desired action  
  
});  
  
</script>  
</body>  
</html>
```

OUTPUT:-



**Login**

Username

Password

e33a1a Refresh

Enter Captcha

Submit Create Delete

## 2.USER ACCOUNT MANAGEMENT :-

### CODE :-

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>User Account Management</title>

  <style>

    body{

      font-family: Arial, sans-serif;

      background-color: #f4f4f4;

      margin: 0;

      padding: 0;

    }

    .container {

      max-width: 600px;
```

```
margin: 50px auto;

padding: 20px;

background-color: #fff;

border-radius: 10px;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

h1 {

text-align: center;

color: #333;

}

form {

display: flex;

flex-direction: column;

margin-bottom: 20px;

}

label {

margin-bottom: 5px;

font-weight: bold;

}

input[type="text"],

input[type="password"],

input[type="email"] {

padding: 10px;

margin-bottom: 15px;

border: 1px solid #ccc;

border-radius: 5px;
```



```
    font-size: 16px;
}
button {
    padding: 10px;
    background-color: #007BFF;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    font-size: 16px;
}
button:hover {
    background-color: #0056b3;
}
.link {
    text-align: center;
    margin-top: 10px;
}
.link a {
    color: #007BFF;
    text-decoration: none;
}
.link a:hover {
    text-decoration: underline;
}
.section {
```

```
        margin-top: 30px;
    }
    .delete-account button {
        background-color: #FF4B4B;
    }
    .delete-account button:hover {
        background-color: #c0392b;
    }
</style>

<script src="https://www.google.com/recaptcha/api.js" async defer></script>
</head>
<body>
    <div class="container">
        <h1>User Account Management</h1>
        <div class="section">
            <h2>Create Account</h2>
            <form action="/submit_account" method="POST">
                <label for="username">Username:</label>

                <input type="text" id="username" name="username" required minlength="3"
maxlength="20" pattern="^[a-zA-Z]+$" title="Username should only contain letters.">

                <label for="email">Email:</label>

                <input type="email" id="email" name="email" required>

                <label for="password">Password:</label>

                <input type="password" id="password" name="password" required minlength="8"
pattern="^(?=.*[a-z])(?=.*[A-Z])(?=.*\d).+$" title="Password must be at least 8 characters
long and contain at least one uppercase letter, one lowercase letter, and one number.">
```

<label for="confirm\_password">Confirm Password:</label>

<input type="password" id="confirm\_password" name="confirm\_password" required minlength="8" pattern="^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d).+\$" title="Password must be at least 8 characters long and contain at least one uppercase letter, one lowercase letter, and one number.">

<div class="g-recaptcha" data-sitekey="YOUR\_SITE\_KEY"></div>

<button type="submit">Create Account</button>

</form>

</div>

<div class="section">

<h2>Manage Account</h2>

<form action="/update\_account" method="POST">

<label for="username">Update Username:</label>

<input type="text" id="username" name="username" required minlength="3" maxlength="20" pattern="^[a-zA-Z]+\$" title="Username should only contain letters.">

<label for="email">Update Email:</label>

<input type="email" id="email" name="email" required>

<label for="password">Update Password:</label>

<input type="password" id="password" name="password" required minlength="8" pattern="^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d).+\$" title="Password must be at least 8 characters long and contain at least one uppercase letter, one lowercase letter, and one number.">

<div class="g-recaptcha" data-sitekey="YOUR\_SITE\_KEY"></div>

<button type="submit">Update Account</button>

</form>

</div>

<div class="section delete-account">

<h2>Delete Account</h2>

<form action="/delete\_account" method="POST">

```
<label for="delete_username">Username:</label>

<input type="text" id="delete_username" name="delete_username" required
minlength="3" maxlength="20" pattern="^[a-zA-Z]+$" title="Username should only contain
letters.">

<div class="g-recaptcha" data-sitekey="YOUR_SITE_KEY"></div>

<button type="submit">Delete Account</button>

</form>

</div>

<div class="link">

  <p><a href="/login">Back to Login</a></p>

</div>

</div>

</body>

</html>
```

#### OUTPUT :-

## User Account Management

### Create Account


Username:

Email:

Password:

Confirm Password:

ERROR for site owner: Invalid site key

  
reCAPTCHA  
[Privacy](#) · [Terms](#)

Create Account

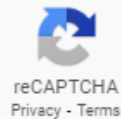
## Manage Account

Update Username:

Update Email:

Update Password:

ERROR for site owner: Invalid  
site key

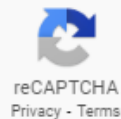


Update Account

## Delete Account

Username:

ERROR for site owner: Invalid  
site key



Delete Account

[Back to Login](#)

### 3.CONTACT US

#### CODE :-

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Contact Us</title>

  <style>

    body{

      font-family: Arial, sans-serif;

      background-color: #f4f4f4;

      margin: 0;

      padding: 0;

    }

    .container{

      max-width: 900px;

      margin: 0 auto;

      padding: 20px;

      background-color: #fff;

      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

    }

    h1{

      text-align: center;

      color: #333;

    }
```

```
.contact-info {  
    margin-bottom: 20px;  
}  
.contact-info h2 {  
    color: #555;  
}  
.contact-info p {  
    margin: 5px 0;  
}  
.contact-info a {  
    color: #007BFF;  
    text-decoration: none;  
}  
.contact-info a:hover {  
    text-decoration: underline;  
}  
.map-container {  
    margin-bottom: 20px;  
}  
.map-container h2 {  
    color: #555;  
}  
iframe {  
    border: 0;  
}  
.contact-form {
```

```
    display: flex;
    flex-direction: column;
}
.contact-form label {
    margin-bottom: 5px;
    font-weight: bold;
}
.contact-form input,
.contact-form textarea {
    margin-bottom: 15px;
    padding: 10px;
    border: 1px solid #ccc;
    border-radius: 5px;
    font-size: 16px;
}
.contact-form button {
    padding: 10px;
    background-color: #007BFF;
    color: white;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    font-size: 16px;
}
.contact-form button:hover {
    background-color: #0056b3;
```



```
}

.message {
    margin-top: 20px;
    font-weight: bold;
    text-align: center;
}

.error {
    color: red;
}

.success {
    color: green;
}

</style>
</head>
<body>
    <div class="container">
        <h1>Contact Us</h1>

        <div class="contact-info">
            <h2>Get in Touch</h2>

            <p>Email: <a
href="mailto:contact@example.com">contact@example.com</a></p>

            <p>Phone: <a href="tel:+1234567890">(123) 456-7890</a></p>

            <p>Address: VIT-AP University, G-30, Inavolu, Beside AP Secretariat, Amaravati,
Andhra Pradesh 522237</p>

        </div>
```

```
<div class="map-container">
```

```
    <h2>Our Location</h2>
```

```
    <iframe
```

```
src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d387140.5071462135!2d80.56464004760036!3d16.516732223512474!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x3a2b7b43f007bb7f%3A0xf7e5c3c7d3f3d64a!2sVIT-AP%20University%2C%20G-30%2C%20Inavolu%2C%20Beside%20AP%20Secretariat%2C%20Amaravati%2C%20Andhra%20Pradesh%20522237!5e0!3m2!1sen!2sin!4v1631031684211!5m2!1sen!2sin"
```

```
    width="100%"
```

```
    height="300"
```

```
    allowfullscreen=""
```

```
    loading="lazy">
```

```
    </iframe>
```

```
</div>
```

```
<form class="contact-form" id="contactForm">
```

```
    <label for="name">Name (max 35 characters):</label>
```

```
    <input type="text" id="name" name="name" maxlength="35" required>
```

```
    <label for="email">Email (max 35 characters):</label>
```

```
    <input type="email" id="email" name="email" maxlength="35" required>
```

```
    <label for="message">Message (max 50 characters):</label>
```

```
    <textarea id="message" name="message" rows="5" maxlength="50"
required></textarea>
```

```
    <button type="submit">Send Message</button>
```

```
    <div class="message" id="formMessage"></div>
```

```
</form>
```

```
</div>
```

```
<script>

document.getElementById('contactForm').addEventListener('submit', function(event) {
    event.preventDefault();

    const name = document.getElementById('name').value.trim();
    const email = document.getElementById('email').value.trim();
    const message = document.getElementById('message').value.trim();
    const formMessage = document.getElementById('formMessage');

    formMessage.textContent = "";
    formMessage.className = 'message';

    if (!name || !email || !message) {
        formMessage.textContent = 'All fields are required.';
        formMessage.className = 'message error';
        return;
    }

    const namePattern = /^[a-zA-Z\s]+$/; // Allow spaces between names
    if (!namePattern.test(name)) {
        formMessage.textContent = 'Name must contain only letters.';
        formMessage.className = 'message error';
        return;
    }

    const emailPattern = /^[^\s@]+@gmail\.com$/;
    if (!emailPattern.test(email)) {
        formMessage.textContent = 'Please enter a valid Email address ending with
@gmail.com.';
        formMessage.className = 'message error';
        return;
    }
});
```

```

    }

    formMessage.textContent = 'Message sent successfully!';

    formMessage.className = 'message success';

    document.getElementById('contactForm').reset();

  });
</script>
</body>
</html>

```


**OUTPUT:-**

## Contact Us

**Get in Touch**

Email: [contact@example.com](mailto:contact@example.com)  
 Phone: (123) 456-7890  
 Address: VIT-AP University, G-30, Inavolu, Beside AP Secretariat, Amaravati, Andhra Pradesh 522237

**Our Location**



**Name:**

**Email:**

#### 4.FINANCIAL DATA AGGREGATION :-

##### CODE :-

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Financial Data Aggregation</title>

  <style>

    body { font-family: Arial, sans-serif; margin: 20px; }

    .form-container { max-width: 600px; margin: 0 auto; padding: 20px; border: 1px solid
#ddd; border-radius: 10px; background-color: #f9f9f9; }

    .form-container h2 { text-align: center; }

    .form-group { margin-bottom: 15px; }

    label { display: block; font-weight: bold; margin-bottom: 5px; }

    input[type="text"], input[type="password"], select { width: 100%; padding: 10px;
border: 1px solid #ccc; border-radius: 5px; }

    button { background-color: #4CAF50; color: white; padding: 10px 20px; border: none;
border-radius: 5px; cursor: pointer; }

    button:hover { background-color: #45a049; }

  </style>

</head>

<body>

<div class="form-container">

  <h2>Aggregate Financial Data</h2>

  <form action="javascript:void(0);" method="post">

    <div class="form-group">
```

```
<label for="bankAccount">Bank Account Number:</label>

<input type="text" id="bankAccount" name="bankAccount" required>

</div>

<div class="form-group">

  <label for="creditCard">Credit Card Number:</label>

  <input type="text" id="creditCard" name="creditCard" required>

</div>

<div class="form-group">

  <label for="investmentAccount">Investment Account Number:</label>

  <input type="text" id="investmentAccount" name="investmentAccount">

</div>

<div class="form-group">

  <label for="dataSource">Choose Financial Data Source:</label>

  <select id="dataSource" name="dataSource">

    <option value="bank">Bank</option>

    <option value="credit_card">Credit Card</option>

    <option value="investment">Investment Account</option>

  </select>

</div>

<button type="submit">Aggregate Data</button>

</form>

</div>

</body>

</html>
```

OUTPUT :-

## Aggregate Financial Data

Bank Account Number:

Credit Card Number:

Investment Account Number:

Choose Financial Data Source:

Aggregate Data

## 5.BUDGETING AND EXPENSE TRACKER

### CODE :-

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Financial and Budget Tracking</title>

  <style>

    body{

      font-family: Arial, sans-serif;

      background-color: #f4f4f4;

      margin: 0;

      padding: 0;

    }

    .container{

      max-width: 600px;

      margin: 50px auto;

      background-color: white;

      padding: 20px;

      border-radius: 10px;

      box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);

    }

    h2{

      text-align: center;

    }

  </style>

</head>

<body>

  <div class="container">

    <h2>Financial and Budget Tracking</h2>

  </div>

</body>

</html>
```



```
label {  
    display: block;  
    margin-top: 10px;  
    font-weight: bold;  
}  
input[type="number"],  
input[type="text"] {  
    width: 100%;  
    padding: 10px;  
    margin: 5px 0 15px 0;  
    border: 1px solid #ccc;  
    border-radius: 5px;  
}  
.summary {  
    background-color: #f9f9f9;  
    padding: 15px;  
    margin-top: 20px;  
    border: 1px solid #ccc;  
    border-radius: 5px;  
}  
.summary p {  
    margin: 5px 0;  
    font-size: 18px;  
}  
button {  
    background-color: #333;
```

```
    color: white;
    padding: 10px 15px;
    border: none;
    border-radius: 5px;
    cursor: pointer;
    width: 100%;
    font-size: 16px;
}
button:hover{
    background-color: #555;
}
.expenses {
    margin-top: 20px;
}
.expenses h3{
    margin-bottom: 10px;
}
.expenses ul{
    list-style-type: none;
    padding: 0;
}
.expenses li{
    background-color: #e9e9e9;
    padding: 10px;
    margin-bottom: 5px;
    border-radius: 5px;
```

```
    display: flex;
    justify-content: space-between;
    align-items: center;
}
.expenses li span {
    font-weight: bold;
}
.delete-btn {
    background-color: #ff4d4d;
    border: none;
    color: white;
    padding: 5px 10px;
    border-radius: 3px;
    cursor: pointer;
}
.delete-btn:hover {
    background-color: #ff1a1a;
}
</style>
</head>
<body>
    <div class="container">
        <h2>Financial & Budget Tracker</h2>
        <label for="income">Monthly Income:</label>
        <input type="number" id="income" placeholder="Enter your income" required>
        <label for="expenseDesc">Expense Description:</label>
```

```
<input type="text" id="expenseDesc" placeholder="Enter expense description"
required>

<label for="expenseAmount">Amount:</label>

<input type="number" id="expenseAmount" placeholder="Enter expense amount"
required>

<button onclick="addExpense()">Add Expense</button>

<div class="summary">

  <p><strong>Total Income:</strong> $<span id="totalIncome">0.00</span></p>

  <p><strong>Total Expenses:</strong> $<span id="totalExpenses">0.00</span></p>

  <p><strong>Remaining Budget:</strong> $<span
id="remainingBudget">0.00</span></p>

</div>

<div class="expenses">

  <h3>Expenses List:</h3>

  <ul id="expensesList"></ul>

</div>

</div>

<script>

  let totalIncome = 0;

  let totalExpenses = 0;

  const expensesList = document.getElementById("expensesList");

  const incomeInput = document.getElementById("income");

  const expenseDescInput = document.getElementById("expenseDesc");

  const expenseAmountInput = document.getElementById("expenseAmount");

  incomeInput.addEventListener("input", function() {

    totalIncome = parseFloat(this.value) || 0;

    updateSummary();
```

```
});  
  
function addExpense() {  
    const expenseDesc = expenseDescInput.value.trim();  
    const expenseAmount = parseFloat(expenseAmountInput.value) || 0;  
    if (expenseDesc === "") {  
        alert("Please enter an expense description.");  
        return;  
    }  
    if (expenseAmount <= 0) {  
        alert("Please enter a valid expense amount.");  
        return;  
    }  
    totalExpenses += expenseAmount;  
    const li = document.createElement("li");  
    li.innerHTML = `${expenseDesc}</span> $$${expenseAmount.toFixed(2)}  
<button class="delete-btn" onclick="deleteExpense(this,  
${expenseAmount})">Delete</button>`;   
    expensesList.appendChild(li);  
    expenseDescInput.value = "";  
    expenseAmountInput.value = "";  
    updateSummary();  
}  
  
function deleteExpense(button, amount) {  
    totalExpenses -= amount;  
    button.parentElement.remove();  
    updateSummary();  
}
```

```

    }

    function updateSummary() {

        document.getElementById("totalIncome").innerText = totalIncome.toFixed(2);

        document.getElementById("totalExpenses").innerText = totalExpenses.toFixed(2);

        document.getElementById("remainingBudget").innerText = (totalIncome -
totalExpenses).toFixed(2);

    }

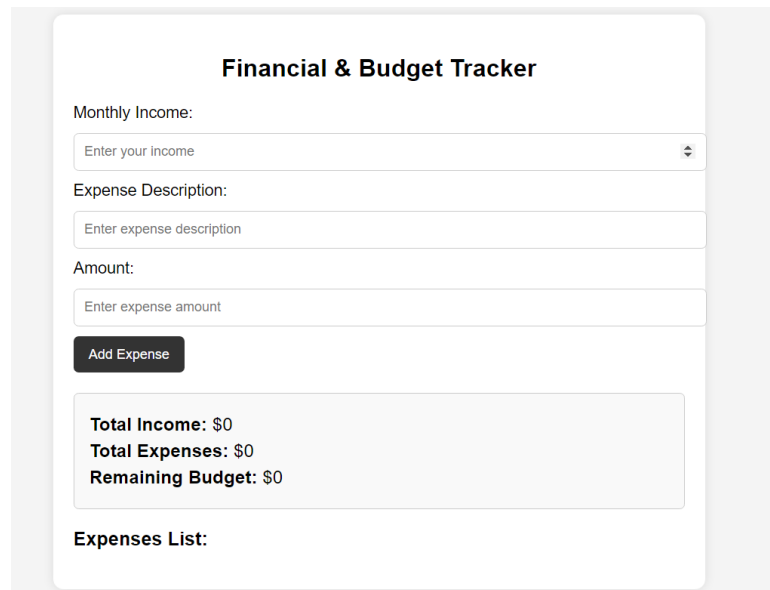
</script>

</body>

</html>

```

### OUTPUT :-



The screenshot displays a web application titled "Financial & Budget Tracker". It features a form with three input fields: "Monthly Income:" (with a placeholder "Enter your income"), "Expense Description:" (with a placeholder "Enter expense description"), and "Amount:" (with a placeholder "Enter expense amount"). Below these fields is a dark "Add Expense" button. A summary box shows "Total Income: \$0", "Total Expenses: \$0", and "Remaining Budget: \$0". At the bottom, there is a section labeled "Expenses List:".

## 6.INVESTMENT MANAGER :-

### CODE :-

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Investment Management</title>

  <style>

    body { font-family: Arial, sans-serif; margin: 20px; }

    .form-container { max-width: 600px; margin: 0 auto; padding: 20px; border: 1px solid
#ddd; border-radius: 10px; background-color: #f9f9f9; }

    .form-container h2 { text-align: center; }

    .form-group { margin-bottom: 15px; }

    label { display: block; font-weight: bold; margin-bottom: 5px; }

    input[type="text"], select { width: 100%; padding: 10px; border: 1px solid #ccc; border-
radius: 5px; }

    button { background-color: #4CAF50; color: white; padding: 10px 20px; border: none;
border-radius: 5px; cursor: pointer; }

    button:hover { background-color: #45a049; }

  </style>

</head>

<body>

  <div class="form-container">

    <h2>Investment Management</h2>

    <form action="javascript:void(0);" method="post">
```

```
<div class="form-group">
```

```
  <label for="investmentType">Investment Type:</label>
```

```
  <select id="investmentType" name="investmentType">
```

```
    <option value="stocks">Stocks</option>
```

```
    <option value="bonds">Bonds</option>
```

```
    <option value="mutual_funds">Mutual Funds</option>
```

```
    <option value="real_estate">Real Estate</option>
```

```
  </select>
```

```
</div>
```

```
<div class="form-group">
```

```
  <label for="amount">Investment Amount:</label>
```

```
  <input type="text" id="amount" name="amount" required>
```

```
</div>
```

```
<div class="form-group">
```

```
  <label for="recommendations">Investment Recommendations:</label>
```

```
  <select id="recommendations" name="recommendations">
```

```
    <option value="low_risk">Low Risk</option>
```

```
    <option value="medium_risk">Medium Risk</option>
```

```
    <option value="high_risk">High Risk</option>
```

```
  </select>
```

```
</div>
```

```
<button type="submit">Submit</button>
```

```
</form>
```

```
</div>
```

```
</body>
```

```
</html>
```



OUTPUT:-

## Investment Management

**Investment Type:**

Stocks

**Investment Amount:**

**Investment Recommendations:**

Low Risk

Submit

## 7.TAX MANAGEMENT :-

### CODE :-

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Simple Tax Management System</title>

  <style>

    body { font-family: Arial, sans-serif; background-color: #f4f4f4; display: flex; justify-
content: center; align-items: center; height: 100vh; margin: 0; }

    .container { background-color: white; padding: 20px; border-radius: 8px; box-shadow:
0 0 10px rgba(0, 0, 0, 0.1); max-width: 400px; width: 100%; }

    h1 { text-align: center; margin-bottom: 20px; }

    label, input { display: block; width: 100%; margin-top: 10px; }

    input { padding: 10px; font-size: 16px; border-radius: 5px; border: 1px solid #ccc; }

    button { width: 100%; padding: 10px; margin-top: 20px; background-color: #5cb85c;
color: white; border: none; border-radius: 5px; cursor: pointer; }

    button:hover { background-color: #4cae4c; }

    #result { margin-top: 20px; padding: 10px; background-color: #f1f1f1; border-radius:
5px; display: none; }

  </style>

</head>

<body>

  <div class="container">

    <h1>Tax Management System</h1>

    <form id="taxForm">
```

```
<label for="income">Annual Income ($):</label>

<input type="number" id="income" placeholder="Enter your income" min="0"
required>

<label for="deductions">Deductions ($):</label>

<input type="number" id="deductions" placeholder="Enter your deductions" min="0"
required>

<label for="taxRate">Tax Rate (%):</label>

<input type="number" id="taxRate" placeholder="Enter tax rate (0-100)" min="0"
max="100" required>

<button type="submit">Calculate Tax</button>

</form>

<div id="result"></div>

</div>

<script>

document.getElementById('taxForm').addEventListener('submit', function(event) {

    event.preventDefault();

    const income = parseFloat(document.getElementById('income').value);

    const deductions = parseFloat(document.getElementById('deductions').value);

    const taxRate = parseFloat(document.getElementById('taxRate').value);

    const resultDiv = document.getElementById('result');

    if (income < 0) {

        resultDiv.textContent = 'Income cannot be negative.';

        resultDiv.style.display = 'block';

        return;

    }

    if (deductions < 0) {

        resultDiv.textContent = 'Deductions cannot be negative.';
```

```
        resultDiv.style.display = 'block';

        return;
    }

    if (deductions > income) {

        resultDiv.textContent = 'Deductions cannot exceed annual income.';

        resultDiv.style.display = 'block';

        return;
    }

    if (taxRate < 0 || taxRate > 100) {

        resultDiv.textContent = 'Tax rate must be between 0 and 100%.';

        resultDiv.style.display = 'block';

        return;
    }

    const taxableIncome = income - deductions;

    const tax = (taxableIncome * taxRate) / 100;

    resultDiv.textContent = 'Your estimated tax is $$' + tax.toFixed(2) + '.';

    resultDiv.style.display = 'block';

    });
</script>
</body>
</html>
```

OUTPUT :-

## Tax Management System

Annual Income (\$):

Deductions (\$):

Tax Rate (%):

Calculate Tax

## 8.REPORTING(FEEDBACK):-

### CODE :-

```
<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <title>Feedback Form</title>

  <style>

    body { font-family: Arial, sans-serif; background-color: #f4f4f4; margin: 0; padding: 0;
display: flex; justify-content: center; align-items: center; height: 100vh; }

    .container { background-color: white; padding: 20px; border-radius: 8px; box-shadow:
0 0 10px rgba(0, 0, 0, 0.1); max-width: 400px; width: 100%; }

    h2 { text-align: center; margin-bottom: 20px; }

    label { margin-top: 10px; font-weight: bold; display: block; }

    input, textarea, select { width: 100%; padding: 10px; margin-top: 5px; border: 1px solid
#ccc; border-radius: 5px; font-size: 16px; }

    textarea { resize: vertical; }

    button { width: 100%; padding: 10px; margin-top: 20px; background-color: #5cb85c;
color: white; border: none; border-radius: 5px; cursor: pointer; }

    button:hover { background-color: #4cae4c; }

    .message { margin-top: 20px; display: none; }

  </style>

</head>

<body>

  <div class="container">

    <h2>Feedback Form</h2>
```

```
<form id="feedbackForm">

  <label for="name">Name:</label>

  <input type="text" id="name" name="name" required placeholder="Your name">

  <label for="email">Email:</label>

  <input type="email" id="email" name="email" required placeholder="Your email">

  <label for="rating">Rating:</label>

  <select id="rating" name="rating" required>

    <option value="5">Excellent</option>

    <option value="4">Good</option>

    <option value="3">Average</option>

    <option value="2">Poor</option>

    <option value="1">Very Poor</option>

  </select>

  <label for="comments">Comments (max 20 words):</label>

  <textarea id="comments" name="comments" rows="4" required placeholder="Your
comments"></textarea>

  <button type="submit">Submit Feedback</button>

</form>

<div id="message" class="message"></div>

</div>

<script>

  document.getElementById('feedbackForm').addEventListener('submit',
function(event) {

    event.preventDefault();

    const name = document.getElementById('name').value;

    const email = document.getElementById('email').value;
```

```
const rating = document.getElementById('rating').value;
const comments = document.getElementById('comments').value;
const messageDiv = document.getElementById('message');
const validDomain = '@gmail.com';
if (!email.endsWith(validDomain)) {
    messageDiv.textContent = Please use an email address that ends with
"${validDomain}";
    messageDiv.style.display = 'block';
    messageDiv.style.color = 'red';
    return;
}
const wordCount = comments.trim().split(/\s+/).length;
if (wordCount > 20) {
    messageDiv.textContent = 'Your comment exceeds the 20-word limit. Please
shorten it.';
    messageDiv.style.display = 'block';
    messageDiv.style.color = 'red';
    return;
}

messageDiv.textContent = Thank you, ${name}! Your feedback has been successfully
submitted.;
messageDiv.style.display = 'block';
messageDiv.style.color = 'green';
});
</script>
</body>
```



</html>**OUTPUT :-**

## Feedback Form

**Name:**

**Email:**

**Rating:**

Excellent

**Comments:**

Your comments

Submit Feedback

## 9.Home Page :-

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <link rel="stylesheet" href="styles.css">

    <title>Project Homepage</title>

</head>

<body>

    <header>

        <h1>Our Project</h1>

        <nav>

            <ul>

                <li><a href="contact.html">Contact Us</a></li>

                <li><a href="feedback.html">Feedback</a></li>

                <li><a href="login.html">Login</a></li>

                <li><a href="registration.html">Register</a></li>

            </ul>

        </nav>

    </header>

    <main>

        <section class="project-info">

            <h2>About Our Project</h2>
```

<p>This AI-powered personal finance manager is designed to help users manage their personal finances by tracking expenses, creating budgets, analyzing spending patterns, and providing useful insights for better financial decisions.</p>

</section>

<section class="team">

<h2>Meet the Team</h2>

<p>B. DHEERAJ</p>

<p>KRISHNA CHAITANYA</p>

<p>N. CHARAN</p>

<p>K. PAVAN KUMAR</p>

<p>N. BHUVANESH</p>

</section>

</main>

<footer>

<p>&copy; 2024 AI Based Personal Finance Manager. All rights reserved.</p>

</footer>

</body>

</html>

OUTPUT :-

## Our Project

- [Contact Us](#)
- [Feedback](#)
- [Login](#)
- [Register](#)

## About Our Project

This AI-powered personal finance manager is designed to help users manage their personal finances by tracking expenses, creating budgets, analyzing spending patterns, and providing useful insights for better financial decisions.

## Meet the Team

B. DHEERAJ

KRISHNA CHAITANYA

N. CHARAN

K. PAVAN KUMAR

N. BHUVANESH

© 2024 AI Based Personal Finance Manager. All rights reserved.

## 10.Registration Form

### Code :-

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Registration Form</title>

    <style>

        body{

            font-family: Arial, sans-serif;

            background-color: #f4f4f4;

            margin: 0;

            padding: 0;

        }

        .container{

            width: 50%;

            margin: 0 auto;

            background-color: #fff;

            padding: 20px;

            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

            margin-top: 50px;

        }

        h2{

            text-align: center;

        }
```

```
label{
    display: block;
    margin-bottom: 8px;
    margin-top: 12px;
}
input[type="text"], input[type="email"], input[type="password"]{
    width: 100%;
    padding: 10px;
    margin-bottom: 20px;
    border: 1px solid #ccc;
    border-radius: 4px;
}
input[type="submit"]{
    width: 100%;
    padding: 10px;
    background-color: #4CAF50;
    color: white;
    border: none;
    border-radius: 4px;
    cursor: pointer;
}
input[type="submit"]:hover{
    background-color: #45a049;
}
</style>
<script>
```

```
function validateForm() {  
    const password = document.getElementById("password").value;  
    const confirmPassword = document.getElementById("confirmpassword").value;  
    if (password !== confirmPassword) {  
        alert("Passwords do not match!");  
        return false;  
    }  
    return true;  
}  
</script>  
</head>  
<body>  
<div class="container">  
    <h2>Registration Form</h2>  
    <form action="/submit_registration" method="post" onsubmit="return validateForm()">  
        <label for="fname">First Name:</label>  
        <input type="text" id="fname" name="firstname" required pattern="[A-Za-z]+"  
title="First name should contain only letters.">  
        <label for="lname">Last Name:</label>  
        <input type="text" id="lname" name="lastname" required pattern="[A-Za-z]+"  
title="Last name should contain only letters.">  
        <label for="email">Email:</label>  
        <input type="email" id="email" name="email" required pattern="[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}" title="Please enter a valid email address.">  
        <label for="password">Password:</label>  
        <input type="password" id="password" name="password" required>
```

```
<label for="confirmpassword">Confirm Password:</label>
<input type="password" id="confirmpassword" name="confirmpassword" required>
<input type="submit" value="Register">
</form>
</div>
</body>
</html>
```

### OUTPUT :-

## Registration Form

First Name:

Last Name:

Email:

Password:

Confirm Password:

Register



## UNIT TEST CASES :-

### CONTACT-US PAGE:

| SI.NO | TestCase   | Expected Result  | Test Result |
|-------|--|--|-------------|
| 1.    | Enter valid name containing only alphabets, Capital or small.  | Software should display “message sent”.<br>Ex:Dheeraj  | Successful. |
| 2.    | Enter invalid Name containing characters other than alphabets. | Software should display error message as “Name should contain only letters”.   | Successful  |
| 3.    | Enter valid email id containg proper format and valid id.      | Software should display “MESSAGE SENT” as output.<br>Ex:dheerajnagaabhishekbandi@gmail.com   | Successful  |
| 4.    | Enter invalid email id like error in format of email etc.      | Software should display error “Please enter a valid email address (e.g., user@gmail.com). Ensure the domain is fully specified”.<br>Ex:bandidheeraj@gma. | Successful  |
| 5.    | For message field if input field is not empty .                | Then software displays message”Message sent”.<br>Ex:Hello guyz   | Successful  |
| 6.    | For message field if input field is empty .                    | Software field displays error of missing field.  | Successful  |

**Login Page :-**

| <b>S.No</b> | <b>Testcases</b>                                      | <b>Expected result</b>                                 | <b>Test result</b> |
|-------------|---|--|--------------------|
| <b>1.</b>   | <b>Enter valid email,password and captcha</b>         | <b>Successful login</b>                                | <b>Successful</b>  |
| <b>2.</b>   | <b>Enter invalid email/password</b>                   | <b>Login failed due to invalid credentials</b>         | <b>Successful</b>  |
| <b>3.</b>   | <b>Enter valid email,password and invalid captcha</b> | <b>Login failed due to entering of invalid captcha</b> | <b>Successful</b>  |
| <b>4.</b>   | <b>Empty fields in email/password</b>                 | <b>Login failed due to occurrence of empty fields</b>  | <b>Successful</b>  |
| <b>5.</b>   | <b>Empty fields in captcha</b>                        | <b>Login failed because of empty field in captcha</b>  | <b>Successful</b>  |
| <b>6.</b>   | <b>Email with no @gmail.com</b>                       | <b>Invalid because no identification of email</b>      | <b>Successful</b>  |

**Home Page :-**

| <b>S.No</b> | <b>Testcases</b>                          | <b>Expected result</b>                | <b>Test result</b> |
|-------------|---|---------------------------------------|--------------------|
| <b>1.</b>   | <b>Press on Contact Us<br/>Hyper Link</b> | <b>Redirecting to Contact Us Page</b> | <b>Successful</b>  |
| <b>2.</b>   | <b>Press on Contact Us<br/>Hyper Link</b> | <b>Bad gateway</b>                    | <b>Successful</b>  |
| <b>3.</b>   | <b>Press on Feed Back<br/>Hyper Link</b>  | <b>Redirecting to Feed Back Page</b>  | <b>Successful</b>  |
| <b>4.</b>   | <b>Press on Feed Back<br/>Hyper Link</b>  | <b>Bad gateway</b>                    | <b>Successful</b>  |
| <b>5.</b>   | <b>Press on Login<br/>Hyper Link</b>      | <b>Redirecting to Login Page</b>      | <b>Successful</b>  |
| <b>6.</b>   | <b>Press on Login<br/>Hyper Link</b>      | <b>Bad Gateway</b>                    | <b>Successful</b>  |
| <b>7.</b>   | <b>Press on Register<br/>Hyper Link</b>   | <b>Redirecting to Register Page</b>   | <b>Successful</b>  |
| <b>8.</b>   | <b>Press on Register<br/>Hyper Link</b>   | <b>Bad Gateway</b>                    | <b>Successful</b>  |

**Feed Back Form :-**

| <b>S.No</b> | <b>Testcases</b>                                  | <b>Expected result</b>  | <b>Test result</b> |
|-------------|---|---|--------------------|
| <b>1.</b>   | <b>Enter valid email</b>                          | <b>Feedback successfully submitted</b><br><br><b>Ex:</b><br><b>bhuvaneshnadella</b><br><b>17@gmail.com.</b> | <b>Successful</b>  |
| <b>2.</b>   | <b>Enter comments words&gt;20</b>                 | <b>Your comment exceeds the 20-word limit. Please shorten it.</b>   | <b>Successful</b>  |
| <b>3.</b>   | <b>Enter valid email,password and words&lt;20</b> | <b>Thank you,</b><br><b>“username”! Your feedback has been successfully submitted.</b>                      | <b>Successful</b>  |
| <b>4.</b>   | <b>Empty fields in username,email</b>             | <b>Please fill out this field.</b>  | <b>Successful</b>  |
| <b>5.</b>   | <b>Empty fields in comments</b>                   | <b>Please fill out this field.</b>  | <b>Successful</b>  |
| <b>6.</b>   | <b>Email with no @gmail.com</b>                   | <b>Please use an email address that ends with "@gmail.com".</b>   | <b>Successful</b>  |

**Registration Form :-**

| <b>S.No</b> | <b>Testcases</b>  | <b>Expected result</b>   | <b>Test result</b> |
|-------------|---|--|--------------------|
| <b>1.</b>   | <b>The fiels shouldn't be empty it shgould contain characters A-Z,a-z</b>                       | <b>Charan12(invalid)</b><br><br><b>Charan(valid)</b>   | <b>successful</b>  |
| <b>2.</b>   | <b>They should be in proper required format</b>   | <b>Charan12(invalid)</b><br><br><b>Charan@gmail.com(valid)</b>   | <b>successful</b>  |
| <b>3.</b>   | <b>In this field shouldn't be empty both the password and the confirm password should match</b> | <b>Password:charan</b><br><br><b>Confirm password:charan12(invalid)</b><br><br><b>Password:charan12</b><br><br><b>Confirm password:charan12(valid)</b> | <b>successful</b>  |

**User Account Management :-**

| Sl.No | TestCase  | Expected Result   | TestResult |
|-------|---|---|------------|
| 1.    | Enter a valid username containing only characters (max=20,min=3)  | Succesful creation of account.<br>Ex:dheeraj                          | Succesful  |
| 2.    | Enter a valid username  | Error display enter username in given format<br>Ex:Dheeraj@1          | Succesful  |
| 3.    | Enter valid email with proper format  | Succesful creation of account<br>Ex;dheeraj@gmail.com                 | Succesful  |
| 4.    | Enter invalid email id.   | Error to input correct format<br>Ex:dheeraj@gmail                     | Succesful  |
| 5.    | Enter valid password minimum 8 length with only letters and characters,(minimum one capital,one small,one number) | Succesful creation of account.<br>Ex:Dheeraj123                       | Succesful  |
| 6.    | Enter invalid password  | Error to input password in correct format<br>Ex:Dheeraj               | Succesful  |
| 7.    | For confirm password enter above password correctly.  | Succesful if same password entered or error to input correct password | Succesful  |

For updating and deleting accounts same constraints as above applied and checked.

**Budget and Expense Tracking :-**

| <b>S.NO</b> | <b>Test Case Description</b>  | <b>Expected Result</b>   | <b>Test Result</b> |
|-------------|---|--|--------------------|
| <b>1.</b>   | <b>Enter valid monthly income and expense details, then click "Add Expense"</b> | <b>Expense is added to the list, and summary values (total income, total expenses, remaining budget) are updated</b> | <b>Successful</b>  |
| <b>2.</b>   | <b>Enter a valid monthly income only without any expense</b>                    | <b>Total income is displayed correctly, with total expenses as zero, and remaining budget calculated correctly</b>   | <b>Successful</b>  |
| <b>3.</b>   | <b>Leave the income field empty and try adding an expense</b>                   | <b>System either prompts the user to enter income or calculates expense without income</b>                           | <b>Successful</b>  |
| <b>4.</b>   | <b>Leave the expense description empty and enter a valid expense amount</b>     | <b>System alerts the user to enter a description or ignores the input</b>  | <b>Successful</b>  |
| <b>5.</b>   | <b>Enter a valid expense description but leave the expense amount empty</b>     | <b>System alerts the user to enter an amount or ignores the input</b>  | <b>Successful</b>  |
| <b>6.</b>   | <b>Enter zero as the expense amount and try to add it</b>                       | <b>System alerts that the expense amount must be greater than zero</b>   | <b>Successful</b>  |
| <b>7.</b>   | <b>Enter a negative amount for expense and try to add it</b>                    | <b>System alerts that the expense amount must be a positive value</b>  | <b>Successful</b>  |

**Investment Manager:-**

| <b>Sl.No</b> | <b>TestCase</b>  | <b>Expected Result</b>  | <b>Test Result</b> |
|--------------|--|---|--------------------|
| <b>1</b>     | <b>Enter a valid investment type (e.g., "Stocks")</b>  | <b>Successful selection of investment type</b>                                      | <b>Successful</b>  |
| <b>2</b>     | <b>Leave investment type blank and submit</b>  | <b>Error message to select an investment type</b>                                   | <b>Successful</b>  |
| <b>3</b>     | <b>Enter a valid investment amount (e.g., "5000")</b>  | <b>Successful entry of investment amount</b>  | <b>Successful</b>  |
| <b>4</b>     | <b>Enter an invalid investment amount (e.g., "abc" or empty input)</b>                                     | <b>Error message to enter a valid numerical investment amount</b>                   | <b>Successful</b>  |
| <b>5</b>     | <b>Enter an amount below minimum required (e.g., "0")</b>  | <b>Error message indicating minimum investment amount requirement</b>               | <b>Successful</b>  |
| <b>6</b>     | <b>Enter amount exceeding maximum allowed (e.g., "10000000" if there's a limit)</b>                        | <b>Error message indicating maximum investment amount limit</b>                     | <b>Successful</b>  |
| <b>7</b>     | <b>Select a valid risk level (e.g., "Low Risk")</b>  | <b>Successful selection of risk level</b>   | <b>Successful</b>  |
| <b>8</b>     | <b>Leave risk level blank and submit</b>   | <b>Error message to select a risk level</b>   | <b>Successful</b>  |
| <b>9</b>     | <b>Complete the form with all valid inputs and submit</b>  | <b>Form successfully submitted, leading to next process or confirmation message</b> | <b>Successful</b>  |
| <b>10</b>    | <b>Complete the form with one or more invalid inputs (e.g., invalid amount and no risk level selected)</b> | <b>Error messages for each invalid input field</b>                                  | <b>Successful</b>  |



**Tax Management :-**

| <b>S.NO</b> | <b>Testcases</b>  | <b>Expected result</b>  | <b>Test result</b> |
|-------------|---|---|--------------------|
| <b>1</b>    | <b>invalid tax percentage<br/>negative values and<br/>&gt;100</b> | <b>Enter valid tax range</b>  | <b>Successful</b>  |
| <b>2</b>    | <b>Enter valid deduction,tax percentage,income</b>                | <b>Success we will get end result</b>                                   | <b>Successful</b>  |
| <b>3</b>    | <b>Deductions&gt;income</b>                                       | <b>Deductions cannot exceed annual income</b>                           | <b>Successful</b>  |
| <b>4</b>    | <b>In valid tax and deductions</b>                                | <b>Deductions cannot exceed annual income and Enter valid tax range</b> | <b>Successful</b>  |
| <b>5</b>    | <b>All valid</b>  | <b>Total amount we get after tax deductions</b>                         | <b>Successful</b>  |

**Aggregating Financial data :-**

| <b>S.NO</b> | <b>Test Case Description</b>  | <b>Expected Result</b>  | <b>Test Result</b> |
|-------------|---|---|--------------------|
| <b>1</b>    | <b>Enter valid bank account, credit card, and investment account details, then click "Aggregate Data"</b> | <b>Form submission is successful; data is accepted, and no error messages are displayed</b> | <b>Successful</b>  |
| <b>2</b>    | <b>Enter a valid bank account number only, leave credit card and investment account fields empty</b>      | <b>Form submission fails with an error message prompting to fill required fields</b>        | <b>Successful</b>  |
| <b>3</b>    | <b>Enter a bank account number with fewer than 8 digits and try submitting the form</b>                   | <b>System alerts that the bank account number must be between 8 and 12 digits</b>           | <b>Successful</b>  |
| <b>4</b>    | <b>Enter a bank account number with more than 12 digits and try submitting the form</b>                   | <b>System alerts that the bank account number must be between 8 and 12 digits</b>           | <b>Successful</b>  |
| <b>5</b>    | <b>Enter non-numeric characters in the bank account number field and</b>                                  | <b>System alerts that the bank account number must contain</b>                              | <b>Successful</b>  |

|    |  |  |            |
|----|--|--|------------|
|    | try submitting the form  | only numeric characters  |            |
| 6  | Enter a valid 16-digit credit card number (without spaces) and try submitting the form           | Form submission is successful; credit card input is accepted without errors        | Successful |
| 7  | Enter a valid 16-digit credit card number (with spaces for grouping) and try submitting the form | Form submission is successful; credit card input is accepted without errors        | Successful |
| 8  | Enter a credit card number with fewer than 16 digits and try submitting the form                 | System alerts that the credit card number must contain exactly 16 digits           | Successful |
| 9  | Enter non-numeric characters in the credit card number field and try submitting the form         | System alerts that the credit card number must contain only numeric characters     | Successful |
| 10 | Enter a valid 10-character alphanumeric investment account number and try                        | Form submission is successful; investment account input is accepted without errors | Successful |

|           |  |   |                   |
|-----------|--|---|-------------------|
|           | <b>submitting the form</b>   |   |                   |
| <b>11</b> | <b>Enter a valid 12-character alphanumeric investment account number and try submitting the form</b> | <b>Form submission is successful; investment account input is accepted without errors</b>                 | <b>Successful</b> |
| <b>12</b> | <b>Enter an investment account number with fewer than 10 characters and try submitting the form</b>  | <b>System alerts that the investment account number must be between 10 and 12 alphanumeric characters</b> | <b>Successful</b> |
| <b>13</b> | <b>Enter an investment account number with more than 12 characters and try submitting the form</b>   | <b>System alerts that the investment account number must be between 10 and 12 alphanumeric characters</b> | <b>Successful</b> |
| <b>14</b> | <b>Enter special characters in the investment account number field and try submitting the form</b>   | <b>System alerts that the investment account number must contain only alphanumeric characters</b>         | <b>Successful</b> |

|    |   |  |            |
|----|---|--|------------|
| 15 | Select "Bank" as the data source and submit with valid account details                          | Form submission is successful; data source selection is accepted without errors                  | Successful |
| 16 | Select "Credit Card" as the data source and submit with valid credit card details               | Form submission is successful; data source selection is accepted without errors                  | Successful |
| 17 | Select "Investment Account" as the data source and submit with valid investment account details | Form submission is successful; data source selection is accepted without errors                  | Successful |
| 18 | Do not select a data source and try to submit the form  | System alerts to choose a financial data source before submitting                                | Successful |
| 19 | Enter valid details in all fields and check credit card input formatting after typing           | Credit card input should be automatically formatted with spaces every 4 digits as the user types | Successful |

GitHub :- <https://github.com/KRISHNACHAITANYAVEJENDLA/AI-based-personal-finance-manager>